REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
31-10-2003	Final Report	

4. TITLE AND SUBTITLE
"A Sensor Management Model Using Simulation-Based Approximate Dynamic Programming"

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
LTC William S. Bland
Dr. Stephen D. Patek
Dr. Sandor Z. Der

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Systems Engineering Dept,
US Military Academy,
West Point, NY

Dept of Systems and Information Engineering,
University of Virginia,
Charlottesville, VA

8. PERFORMING ORGANIZATION REPORT NUMBER

Sensors and Electron Devices Directorate,
Army Research Laboratory,
Adelphi, MD

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Unlimited Distribution

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This paper presents a new approach to sensor management of distributed sensor networks (DSNs). Given the current proliferation of remote sensors and their inherent resource constraints, DSN managers face a growing problem of managing the tradeoff between DSN performance and resource consumption. Our model, the Sensor Network Optimal OPerations Simulator, or SNOOPS, addresses this tradeoff by identifying a DSN control strategy that reaches an acceptably certain representation of the search region while minimizing operating costs.

The core of the SNOOPS model is an approximate dynamic programming (ADP) process that uses simulation-based policy iteration to identify an efficient DSN control strategy. Results indicate that the SNOOPS-recommended DSN control strategy improves the efficiency of DSN operations by up to 47 percent over the Base Policy of activating all sensors.

In addition to determining efficient DSN control strategies, our model also provides a research base to (1) investigate the fusion of observations from disparate sensors, (2) demonstrate the use of non-imaging sensors to provide adequate situational awareness where precision emplacement of more-capable sensors is not possible, and (3) develop operational concepts to integrate DSN operations with user needs.

15. SUBJECT TERMS
Sensor Management, Approximate Dynamic Programming, Policy Iteration, Simulation, Distributed Sensor Network

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			LTC William S. Bland
Unclassified	Unclassified	Unclassified	UL	26	19b. TELEPHONE NUMBER *(include area code)* 845-938-5181

A Sensor Management Model Using Simulation-Based Approximate Dynamic Programming

LTC William S. Bland
Stephen D. Patek
Sandor Z. Der

Descriptor List

Sensor Management
Distributed Sensor Network
Approximate Dynamic Programming
Policy Iteration
Simulation

A Sensor Management Model Using Simulation-Based Approximate Dynamic Programming

LTC William S. Bland
U.S. Military Academy
West Point, New York
william.bland@usma.edu

Stephen D. Patek
University of Virginia
Charlottesville, Virginia
patek@virginia.edu

Sandor Z. Der
Army Research Laboratory
Adelphi, Maryland
sder@arl.army.mil

71st MORS Symposium
Working Group 6 - C4ISR
31 October 2003

Abstract

This paper presents a new approach to sensor management of distributed sensor networks (DSNs). Given the current proliferation of remote sensors and their inherent resource constraints, DSN managers face a growing problem of managing the tradeoff between DSN performance and resource consumption. Our model, the Sensor Network Optimal OPerations Simulator, or SNOOPS, addresses this tradeoff by identifying a DSN control strategy that reaches an acceptably certain representation of the search region while minimizing operating costs.

The core of the SNOOPS model is an approximate dynamic programming (ADP) process that uses simulation-based policy iteration to identify an efficient DSN control strategy. Results indicate that the SNOOPS-recommended DSN control strategy improves the efficiency of DSN operations by up to 47 percent over the Base Policy of activating all sensors.

In addition to determining efficient DSN control strategies, our model also provides a research base to (1) investigate the fusion of observations from disparate sensors, (2) demonstrate the use of non-imaging sensors to provide adequate situational awareness where precision emplacement of more-capable sensors is not possible, and (3) develop operational concepts to integrate DSN operations with user needs.

Introduction

Traditional sensor networks have been in use for years for military and civilian surveillance applications, as well as for electronic, economic, medical, and hazard detection systems. These traditional sensor networks used relatively simple sensors that were connected to permanent infrastructure, with virtually unlimited power and communications resources. Without strict resource constraints, the DSN sensor management problem was trivial – keep all the sensors active all the time, continuously sampling the environment and reporting observations, referred to in this paper as the Base Policy. Use of the Base Policy led to a greater emphasis on sensor fusion than on sensor management.

Recently, however, the types of sensors used in these sensor networks have begun to evolve, primarily due to advances in integrated circuit, computing, and communications technologies (Pottie and Kaiser 2000). These technological advances have resulted in the ability to build advanced micro-electromechanical systems (MEMS), which have in turn enabled the development of more capable, remote sensors with a wireless communication capability. These remote sensors, commonly referred to as unattended ground sensors (UGS), consist of a variety of sensor technologies that are packaged for deployment and perform the mission of remote target detection, location, and/or recognition (Srour 1999),

Networks of intelligent, disparate sensors (like UGS) that are distributed spatially and geographically are generally referred to as Distributed Sensor Networks, or DSNs. The incorporation of UGS has vastly expanded the capabilities of DSNs versus traditional sensor networks by facilitating rapid deployment and reconfiguration in largely unconstrained arrangements (Clare et al. 1999).

However, these same UGS have also introduced new battery power and communications bandwidth constraints. While significant accomplishments have been achieved in developing sensor-level approaches to address these constraints, there still exists a need for system-level management of individual DSN assets.

In the remainder of this paper, we formally define the DSN sensor management problem, identify the Test Bed Scenario used in our research, briefly review related DSN sensor management research, describe our general approach and methodology for executing this approach, identify our modeling approach, define the mathematical formulation for our solution technique, present some interesting results, and offer a plan for continued research.

Problem Definition

In its simplest form, DSN sensor management is fundamentally a dynamic resource-allocation problem (Malhotra 1995), where we must balance DSN performance (measured in terms of target detection and localization) against resource consumption (measured in terms of power usage). Solving this problem necessitates developing a sequential decision process for providing control over a sensor network where the penalties and rewards for actions are only revealed over time.

The sequential aspect of this process lends itself to interpretation as a closed-loop feedback control system, as described by Bertsekas (2000) in Figure 1.

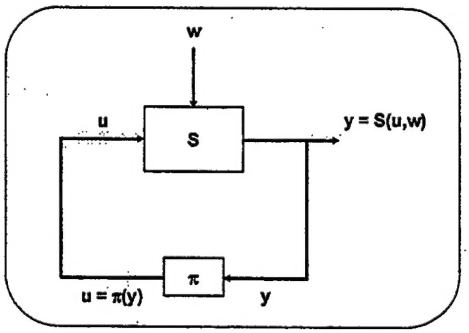

Figure 1: Closed-Loop Feedback Control System

In this figure, u depicts the Input, w depicts the random Disturbance, S is the System Function, y depicts the Output, and π is the Feedback Controller. The term $u = \pi(y)$ indicates that each Input is a function of the previous Output.

We can easily interpret the DSN sensor management problem in terms of a closed-loop feedback control system. Initial control instructions direct specific sensors to take measurements of the environment. These measurements, or observations, are translated into sensor reports, which are fused using a Bayesian approach to provide an updated representation of the search region. This updated representation is then used by the Feedback Controller to determine the next iteration of sensor control instructions.

Using this interpretation, our objective in addressing the DSN sensor management problem is to develop a strategy for the Feedback Controller that allows us to detect and locate any objects of interest in the search region while conserving scarce resources. In other words, we need to develop a model that individually tasks DSN sensing assets in such a manner as to reach an acceptably certain representation of the search region while minimizing DSN operating costs, measured in terms of power usage.

Test Bed Scenario

To provide a specific context for the DSN sensor management problem, we created a Test Bed Scenario, described below.

Search Region. The search region for the Test Bed Scenario consisted of the three-kilometer by three-kilometer terrain box depicted in Figure 2.

Objective. The objective for the Test Bed Scenario consisted of using the DSN to detect and locate any objects of interest located within the search region while minimizing DSN operating costs in order to extend the life of the DSN.

Sensor Network. The DSN for the Test Bed Scenario was fairly dense, consisting of 35 individual sensing nodes to cover the search region. We used only one sensor type to simplify the sensor fusion problem, selecting acoustic sensors since they are the most common non-imaging sensors in use today (Hopkins et al. 2000). The acoustic sensors modelled in our scenario consisted of a circular array of microphones that are capable of providing a line of bearing estimate to a detected object of interest.

Figure 2: Terrain Box

We modelled the sensors with a "self-locating" capability, a realistic expectation as there are currently several research efforts ongoing to enable self-localization of a network of sensors (e.g., see (Moses et al. 2002)). To simplify the problem, we chose to establish a fixed cluster topology, with the 35 sensors organized into four logical clusters, with from 6 to 12 sensors in each cluster, as shown in Figure 3.

In this configuration, each sensor is subordinate to a cluster head, which is in turn subordinate to the sensor network controller.

We modelled the sensors with onboard power management tools that would permit two operating modes. In the "Sleep" mode, a sensor operates at the lowest pos-

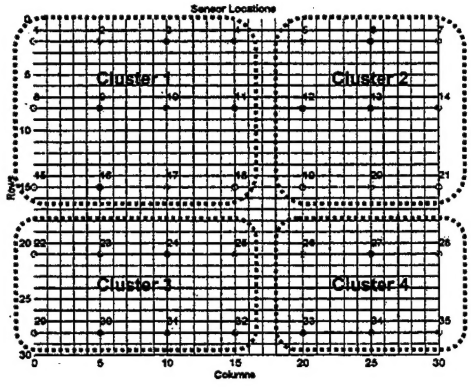

Figure 3: Sensor Location and Cluster Topology

sible power consumption level and is just awaiting instructions to begin sensing the environment. Once awakened, the sensor operates in an "Active" mode. In this mode, the sensor actively observes the environment, expending higher amounts of energy. If the sensor detects an object of interest, whether a true detection or false alarm, it will then expend an additional amount of energy to transmit a sensor report.

Related Work

In recent years, there has been an increased level of effort focused on developing methodologies that individually task DSN sensing assets. Three of the more promising DSN sensor management approaches being developed include Bayes Risk minimization, information-theory, and dynamic programming (DP).

Bayes Risk Minimization. In this approach, the objective is to apply available sensing assets so as to minimize the conditional expectation of the Bayes average risk with respect to a pre-defined loss functional. In effect, sensors are selected to reduce the risk of making an incorrect detection or localization decision.

In Sinno et al. and Cochran et al. (1999), the search region is divided into a number of disjoint cells and a hypothesis is established for each cell that a target is present in that particular cell. Bayes Rule is applied to the prior probabilities of the hypotheses and the sensors' probabilities of detection and false alarm to calculate the posterior probabilities of each hypothesis, given a particular test and its outcome. Using these posterior probabilities, the conditional expectation of the Bayes average risk is determined before any test is actually run and the action resulting in the minimum expected Bayes Risk is selected.

While this sort of a closed-loop feedback control policy is useful for its stated purpose, it fails to address two of our critical DSN sensor management requirements: reducing uncertainty in the search region representation and minimizing resource consumption.

Information Theory. In this approach, the objective is to apply available sensing assets so as to reduce the uncertainty in the state and hence produce a quantifiable amount of information. McIntyre (1996) postulates that sensor observations produce potential information gains that may reduce (i) uncertainty of location of undetected targets, (ii) uncertainty associated with the estimate of a target's state vector, or (iii) uncertainty associated with target identity.

McIntyre (1998) and McIntyre and Hintz (1997) present models that schedule sensor usage based on Shannon's definition of entropy as a measure of potential information gain. The models use the measure of potential information gain to determine whether to use sensor resources to track targets already in track or to search for new targets,

and then to decide which sensor to use.

Kastella (1997) and Schmaedeke (1993) present models that schedule sensor usage based on the Kullback-Leibler discrimination information function as a measure of potential discrimination gain. The models estimate the expected discrimination gain for the observation of each sector of the search region and then activate sensors in such a way as to maximize the expected discrimination gain.

This closed-loop feedback control policy shows promise as a DSN sensor management policy since it is focused on reducing the uncertainty in the search region representation. However, it chooses to perform the best action at each particular sensing iteration, without regard for the evolution of the scenario. This may not be desirable since one may prefer to perform actions that are not optimal in the information-theoretic sense but are superior in terms of mission success (Malhotra 1995).

Dynamic Programming. In this approach, the objective is to apply the sensing assets so as to optimize DSN performance within specified resource constraints. Castañon (1995) applies DP to the sensor management problem, characterizing it as a dynamic sequential hypothesis testing problem, with hypotheses associated with specific subregions of the search region. His model selects sensors to maximize the composite probability of identifying the correct hypothesis at the end of a fixed horizon.

In subsequent work, Castañon (1997) uses DP to specifically address the problem of classifying a known number of unknown objects. He again formulates the problem as a sequential hypothesis testing problem, where the hypotheses are associated with specific classifications of the objects. His model finds a decision rule that minimizes the expected total cost over all admissible decision rules, subject to sensor use constraints.

A DP-based closed-loop feedback control policy appears to be ideally suited for our definition of DSN sensor management. In fact, Bertsekas (2000) claims that "DP is the only general approach for sequential decision-making" because of its guaranteed convergence to the optimal solution (given the correct cost-to-go's) and its immunity to noise. Additionally, Schmaedeke (1993) declares that "One of the most promising methods to provide the sequential decision process necessary to develop control instructions is approximate dynamic programming (ADP)."

General Approach

Our approach to address the DSN sensor management problem consists of developing an ADP-based closed-loop feedback control process that identifies an efficient DSN control strategy. The framework we used to develop this closed-loop feedback control process is depicted in Figure 4.

In Step 1 of this framework, we decide which sensors to activate for the particular sensing iteration, using an ADP construct described below. In Step 2, the active sensors operate, taking readings and passing reports up the DSN hierarchy, as defined in the cluster topology. In Step 3, we fuse the reports submitted by the active sensors to update our representation of the search region. Finally, in Step 4, we determine whether our current representation of the search region is adequate enough to terminate the DSN mission. If not, we return to Step 1 and repeat

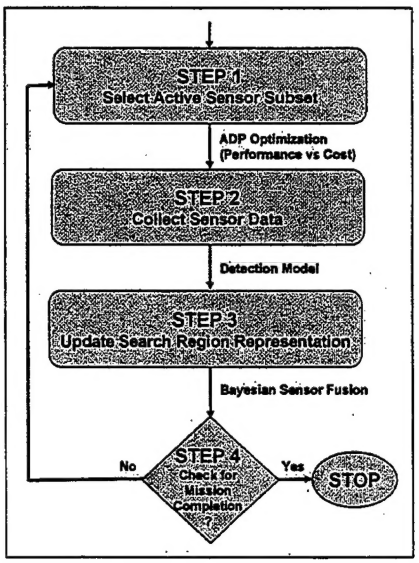

Figure 4: Sensor Management Framework

the cycle until either an object of interest is located or we are certain enough that no object is present.

The ADP optimization process balances both present and expected future costs while working towards an ultimate objective. At each stage of the problem, we use Monte Carlo simulation to implement a one-step lookahead version of a rollout policy to develop approximations for the transition costs and cost-to-go functions for a number of promising control actions. We then use these approximations to select the best candidate action, execute the selected action, and then proceed to the next stage, continuing this process until we achieve the DSN objective.

Methodology

Our methodology for implementing and validating the approach described above consists of executing the following tasks:

- DSN Simulation Model Development
- DP Formulation
- Policy Comparison Analysis

These three tasks are summarized below and described in detail in subsequent sections of the paper.

DSN Simulation Model Development. Since there was no actual DSN available with which to experiment, we developed our own DSN simulation model. Our model, the Sensor Network Optimal **OP**erations Simulator, or SNOOPS, serves two primary purposes. First, it simulates the performance of any static or dynamic stationary sensor control policy. Second, it implements the aforementioned ADP process to identify a "near-optimal" sensor control policy (hereafter referred to as the SNOOPS Policy).

DP Formulation. We used the DP model for dynamic search suggested in Castañon (1995, 1997) and revisited in Patek (2001) as the basis for the mathematical formulation of our sensor management process. The cited works assume that a sensor searches only one cell at a time and that false alarms will not occur. In our model, we extend these results in a number of ways. First, we consider searches where multiple cells may be searched simultaneously. Second, we consider searches which can produce false alarms. Finally, we extend the formulation from a finite to an infinite horizon.

Policy Comparison Analysis. Using the SNOOPS Model, we quantified the performance of the Base Policy and SNOOPS Policy. We collected output data that included the total cost to reach termination, the number of stages required to reach termination, and the amount of computation time required to reach termination. We also included the performance of various other sim-

ple dynamic sensor control policies to provide additional performance comparisons.

Modelling Approach

The SNOOPS DSN simulation model makes use of three fundamental modelling components: a Detection Model, a Sensor Fusion Model, and a Bayesian Model.

Detection Model. In the SNOOPS model, each active sensor makes a binary decision, y, between two hypotheses for each cell observed: H_1 that an object of interest is present in the cell and H_0 that an object of interest is not present in the cell. The likelihood of declaring whether H_0 or H_1 is true is dictated by the sensor's performance capabilities.

In standard signal detection theory, α corresponds to the probability of a "False Alarm" (or false positive) and β corresponds to the probability of a "Miss" (or false negative). Conversely, the probability of a "Detection" (or true positive) is $(1-\beta)$ and the probability of a "Quiet" outcome (or true negative) is $(1-\alpha)$.

SNOOPS uses the curves depicted in Figures 5 and 6 to represent sensor performance characteristics.

Figure 5 represents the probability of detection (PD) curve used to determine $(1-\beta)$ while Figure 6 represents the probability of false alarm (PF) curve used to determine α. Both of these curves are a function of sensor-target (S-T) distance.

If the sensor detects an object of interest (either a true or false positive), it submits a sensor report. SNOOPS handles directional and non-directional sensors differently. Directional sensors (e.g., acoustic) are able to

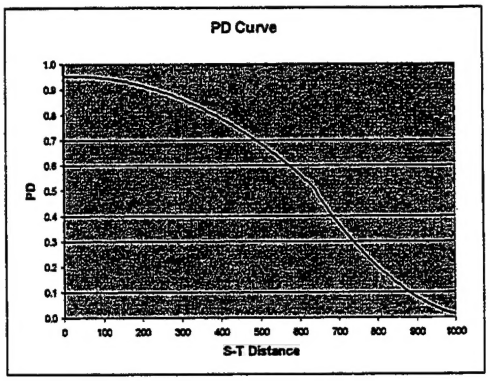

Figure 5: PD Curve

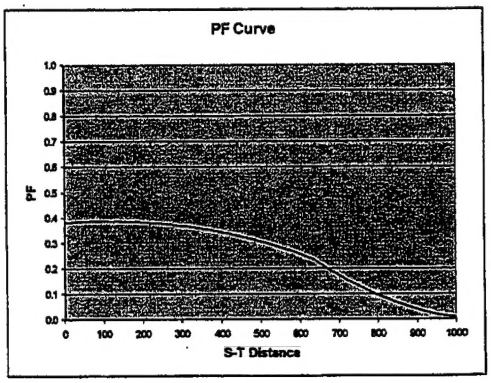

Figure 6: PF Curve

provide a bearing to the target, reported in the range (0, 360), with 360 degrees indicating due North. The result of a detection will be a "Report Cone," a region extending out to the sensor's maximum range and centered on the reported bearing to the target. The width of this cone is determined by the sensor capabilities. A Report Cone is graphically depicted in Figure 7.

Non-directional sensors (e.g., seismic), on the other hand, are unable to provide a bearing to the target. The result of a detection will be a "Report Disk," a region centered on the sensor location and extending to its maximum range. A Report Disk is graphically depicted in Figure 8.

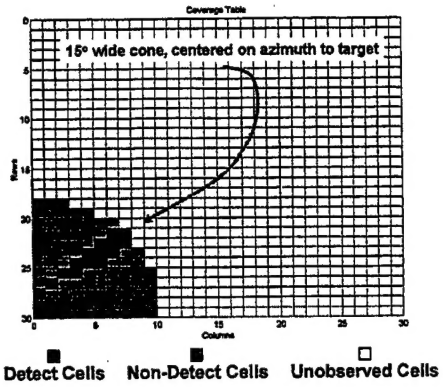

Figure 7: Report Cone

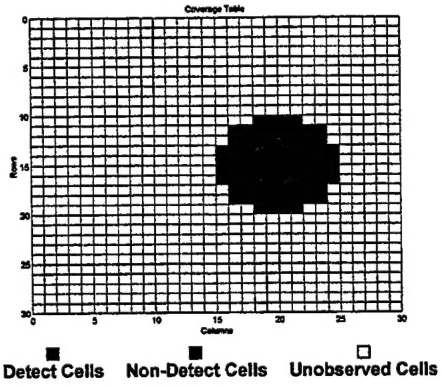

Figure 8: Report Disk

Sensor Fusion Model. SNOOPS uses a distributed fusion approach, with sensor fusion occurring at multiple locations throughout the DSN. Each cluster head fuses the input from its subordinate sensors and then forwards a fused cluster report to the DSN controller. The DSN controller than fuses these inputs to develop a global inference. This sharing of the computational burden throughout the sensor network provides several benefits, including reduced power consumption and bandwidth.

While the sensor reports are created in response to a binary local decision, each of the fusion centers in the SNOOPS model uses a fusion of probabilities scheme to fuse the reports. In this scheme, detection probabilities are used instead of actual observation vectors or local decisions. These detection probabilities are derived from the previously described sensor performance curves. Krzysztofowicz and Long (1990) claim that the fusion of probabilities scheme offers high performance and flexibility for distributed multi-sensor systems, has modest requirements for communications channels, and is appropriate for situations where observations naturally occur in the form of detection probabilities.

Bayesian Model. The centerpiece of the fusion of probabilities fusion scheme described above is an implementation of Bayes Theorem, which describes how to develop inferences based on prior evidence and current observations. In the SNOOPS model, we assume that there exists a current belief about the hypothesis that an object of interest is present in cell i prior to receiving any sensor reports. This prior, or unconditional, belief about the probability of cell i containing a target is $p(H_1)$, and our prior belief that cell i does not contain a target is then $p(H_0) = 1 - p(H_1)$.

Consider y as a binary observation returned by a sensor after searching cell i, with the values of y being either 0 or 1. The case where $y = 1$ indicates that an object of interest was detected in cell i, and $y = 0$ indicates that no object of interest was detected in cell i. The likelihood of observing y given the presence (or absence) of an object of interest is then $p(y|H_1)$ (or $p(y|H_0)$). These detection probabilities are derived from the appropriate sensor performance curves.

The sensor fusion problem becomes: Given y, determine the posterior probability of cell i containing an object of interest, or in other words, find $p(H_1|y)$. In general, we can use

Bayes Theorem to find $p(H_1|y)$ as follows:

$$p(H_1|y) = [p(y|H_1) \times p(H_1)]/p(y),$$

with $p(y) = p(y|H_1)p(H_1) + p(y|H_0)p(H_0)$. For our model, we have formulated specific implementations of Bayes Theorem, which we will describe in detail in the System Dynamics section of this paper.

DP Formulation

We begin the mathematical formulation of our sensor management approach by establishing some initial notation. We then define our model in terms of normal DP elements: states, control actions, disturbances, system dynamics, cost structure, objective function, and finally, our ADP solution approximation.

Initial Notation. We introduce the following notation that will be used throughout the remainder of this paper (with more detailed explanations provided in subsequent sections):

Let E denote the environment (search region) consisting of C search locations, or cells. Each cell $c_i, i = 1, 2, \ldots, C$, represents a unique portion of the search region and $E = \bigcup_{i=1}^{C} c_i$.

Associated with each cell c_i there is a property, P^i, defined as "cell i contains an object of interest."

Let x^i denote the probability that property P^i is true, given all available information. Let $x = \{x^1, x^2, \ldots, x^C\}$ be the vector of current probabilities associated with each of the cells in the search region.

Let $t = 0, 1, \ldots$ denote the stage (sensing iteration) of the DSN sensor management problem. In general, the DSN sensor management process proceeds indefinitely until we reach an acceptable level of certainty about the search area representation (Malhotra 1995). For this reason, we cannot assume that there is a natural (fixed, finite) time horizon, a priori.

Let $G \in \Re$ denote the number of targets in the search region.

Let S denote the set of sensors available to observe the search region. Each sensor $s = 1, 2, \ldots, m$ can observe a portion of the search region, the set of cells E^s. It is important to note that the sensor footprints $E^s, s = 1, 2, \ldots, m$ are not mutually exclusive sets (i.e., each cell i may exist in multiple sensor footprints) and are constant for all stages t.

Let u denote a test that is applied to the search region, where $u \in U$, the set of all possible tests. Each test u corresponds to a different subset of the total available sensors S.

Let S^u denote the set of sensors that are included in test u. The subset of cells observed during test u will depend on the sensors included in the test and is denoted by E^u, where $E^u = \bigcup_{s \in S^u} E^s$.

Let $y^{i,s}$ denote the local binary decision made at sensor s after observing cell i. The value of this decision corresponds to a tentative decision concerning the presence of an object of interest in the cell. The binary decision can take values in $\{0, 1\}$, with probabilities described as follows for each sensor s included in S^u:

$$P(y^{i,s} = 1|\text{sensor } s \text{ active}) = \begin{cases} \alpha^{i,s} & : \text{if } P^i \text{ is not true} \\ 1 - \beta^{i,s} & : \text{if } P^i \text{ is true} \end{cases} \quad (1)$$

$$P(y_t^{i,s} = 0|\text{sensor } s \text{ active}) = 1 - P(y^{i,s} = 1|\text{sensor } s \text{ active}) \quad (2)$$

When $y^{i,s} = 1$, we say that cell i is a member of D^s, the subset of cells in E^s that are considered Detect cells. When $y^{i,s} = 0$, we say that cell i is a member of N^s, the subset of cells in E^s that are considered Non-Detect cells. The sets D^s and N^s are mutually exclusive and $E^s = D^s \cup N^s$.

Let z denote the observation made by the current set of active sensors, S^u, where $z \in Z$, the observation space. The observation z consists of the vectors D^s and N^s, for each $s \in S^u$. For the example shown in Figure 7, where a singe sensor s was activated, the resultant observation z consisted of the vectors $N^s = \{\text{set of gray cells}\}$ and $D^s = \{\text{set of black cells}\}$.

Problem Types. We consider two types of DSN sensor management problems, Type I and Type II, each related to a different assumption regarding the interdependence of the hypotheses. In the following definitions, we borrow the terminology of "exclusive" and "independent" hypotheses from Castañon (1995).

Type I Problem. In this type of problem, we have a single C-ary hypothesis testing problem, where for each cell i there exists a hypothesis H^i that property P^i is true. For these problems, we assume that these hypotheses are "exclusive" (i.e., exactly one hypothesis $\{H^0, H^1, \ldots, H^C\}$ is true). This would correspond to a scenario with exactly one object of interest in the search region. For Type I problems, the component values of x will necessarily sum to unity for each stage of the problem.

Type II Problem. In this type of problem, we have C independent hypothesis testing problems, one for each cell i. In each of these hypothesis testing problems, there exists the hypothesis H^i that property P^i is true and the null hypothesis that there is nothing of interest in cell i. For these problems, we assume that the hypotheses are "independent" (i.e., multiple hypotheses $\{H^0, H^1, \ldots, H^C\}$ may be true). This would correspond to a scenario with multiple objects of interest in the search region. This is also the most appropriate category for a scenario where the number of objects in the search region G is unknown and will be the most frequent problem type encountered. For Type II problems, the component values of x will not sum to unity in general.

States. In our interpretation of the DSN sensor management problem our decisions regarding sensor control are based on the results of noisy sensor observations. While we can safely assume that one sensor's observations will not have an impact on another sensor's observations, it is probably less safe to assume that no sensor observations will be correlated. For example, several sensors will likely report on the presence of a target (or loud natural phenomenon), given it is within each of their footprints. While the sensor observations may not be quite independent, we will treat them as such for the practical purposes of our Test Bed Scenario.

Once received, these sensor observations are stored in the information vector I_t, where $I_t = (z_0, z_1, \ldots, z_t, u_0, u_1, \ldots, u_{t-1})$ for $t =$

$0, 1, \ldots$ and $I_0 = z_0$. The information vector defines the information available after stage t, including the sequence of tests conducted and observations received through stage t.

In our solution to the DSN sensor management problem, we are looking for a rule that tells us the control u_t to be applied for every possible information vector I_t. As new observations are added at each stage t, the dimension of the information vector I_t increases accordingly. Since it is necessary to apply the DP algorithm over the entire space of I_t, solution of this problem may be very complex and computationally impossible for large values of t.

To simplify the problem, we need a function that is of smaller dimension than I_t, yet summarizes all the essential content of I_t as far as control is concerned. As shown in Bertsekas (2000), a useful sufficient statistic is the conditional probability distribution of the state. In our case, this sufficient statistic is represented by x_t, the conditional probability distribution at stage t, where $x_t = \{x_t^1, x_t^2, \ldots, x_t^C\}$, such that x_t^i is the conditional probability that property P^i is true, given the information available at stage t, or $P(P^i = 1|I_t)$.

The state space X is a convex set which represents the set of feasible states for the problem. For Type I problems, X can be considered to be the n-dimensional unit simplex since the x_t^i values are constrained to sum to unity, while for Type II problems, X is much larger since the individual x_t^i values are not constrained in the same manner.

Initial State. Our initial state is the prior distribution, $x_0 = \{x_0^1, x_0^2, \ldots, x_0^C\}$, such that x_0^i is the initial probability that the property P^i is true. This prior distribution consists of subjective probabilities which measure the user's degree of belief in the presence of an object of interest within each cell before applying any tests. For Type I problems, these values are normalized so that the elements of x_0 will sum to unity.

For the Test Bed Scenario simulation runs, SNOOPS used the prior distribution graphically represented in Figure 9 as the initial state. In this figure, the color of each cell represents the value of $p(H_1)$ for that cell, with light colors corresponding to low probabilities and dark colors corresponding to high probabilities.

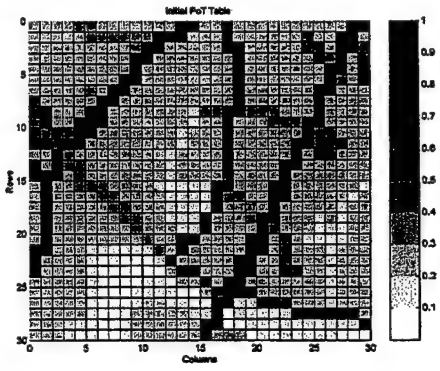

Figure 9: Initial Probabilities

It is easy to correlate the prior probabilities depicted in Figure 9 with the terrain box shown in Figure 2. The highest probabilities correspond to road and trail networks while the lowest probabilities correspond to waterways and other major obstacles.

Termination States. A key feature of our formulation is that the decision process proceeds indefinitely until we reach certainty (or near certainty) about each of the hypotheses. To capture termination within this framework, we assume that there exists a space of special termination regions Ω_γ, where γ is the tolerance that describes our required degree of certainty.

Just as optimal solutions in linear programming exist at extreme points in the feasible region, termination regions exist around the extreme points or vertices of the state space in our problem. For Type I problems, the vector $x = (0, \ldots, 0, 1, 0, \ldots, 0) \in X$ (with the 1 in the i-th component) corresponds to certainty about the property P^i. For Type I problems, there will be C such termination regions.

For Type II problems, the vector $x_t = (0, \ldots, 0, 1, 0, \ldots, 0, 1, 0, \ldots, 0) \in X$ (with a 1 in the i-th and j-th components) corresponds to certainty about the properties P^i and P^j. A similar construct exists for Type II problems where more than two objects of interest exist within the search region. Finally, the vector $x_t = (0, \ldots, 0) \in X$ corresponds to certainty that the null hypothesis is true (i.e., there are no objects of interest in the search region). For Type II problems, there will be 2^C such termination regions.

A termination state $x_\Omega \in \Omega_\gamma$ is described as a state that satisfies the requirement that each $x_t^i, i = 1, \ldots, C$ lie within the range $(0, \gamma)$ or $((1-\gamma), 1)$. The termination states are absorbing, and once the system reaches a termination state it remains there at no further cost. We point out that the case where $\gamma = 0$ is reasonable whenever there exist all-powerful tests that can locate an object of interest with certainty.

Control Actions. At each stage t of the problem, we choose a test $u_t \in U$ that corresponds to a specific subset of the total available sensors. The control space U consists of all possible combinations of sensors, which translates to a total of $2^S - 1$ unique tests from which to choose, given there are S sensors available.

Disturbances. The random disturbance w_t is an element of a space W_t, $t = 0, 1, \ldots$ and is characterized by a probability measure defined on a collection of events in W_t. This probability measure may depend explicitly on x_t and u_t but not on values of prior disturbances $w_{t-1}, \ldots, w_0$. These disturbances describe the stochastic nature of the DSN sensor management problem, aspects of the problem that are not controllable.

System Dynamics. The system dynamics are described by a transition function f_t, a function that describes the evolution of the system from state x_t to state x_{t+1}. The transition process works as follows: at stage t the system is in state x_t, we make a decision to apply the sensors in test u_t, and the system incurs a random disturbance w_t, driving it to state x_{t+1}. The system equation that describes this transition is as follows:

$$x_{t+1} = f_t(x_t, u_t, w_t), \quad t = 0, 1, \ldots . \quad (3)$$

The function f_t effects the transition from state x_t to state x_{t+1}, where $x_{t+1} = (x_{t+1}^1, x_{t+1}^2, \ldots, x_{t+1}^n)$. This state transition is effected through a Bayesian fusion of the prior state x_t with the local decisions $y_t^{i,s}$ (and corresponding probabilities) resulting from applying test u_t. The specific evolution of the posterior conditional probabilities is different for Type I and Type II problems, as described below.

Conditional Probability Evolution - Type I Problems. For Type I problems, measurements resulting from searching cell i will affect all of the conditional probabilities x_{t+1}. For this reason, the conditional probability evolution for Type I problems is much more complicated than for Type II problems.

For each cell $c_i \in D_t^s$ (a Detect Cell), the conditional probability evolution is given below:

$$x_{t+1}^i = \left[x_t^i(1-\beta^{i,s}) \times \prod_{j \in D_t^s -} (\alpha^{j,s}) \times \prod_{k \in N_t^s} (1-\alpha^{k,s}) \right] \div \left[\sum_D + \sum_N + \sum_L \right] \quad (4)$$

For each cell $c_i \in N_t^s$ (a Non-Detect Cell), the conditional probability evolution is given below:

$$x_{t+1}^i = \left[x_t^i(\beta^{i,s}) \times \prod_{c_j \in D_t^s} (\alpha^{j,s}) \times \prod_{c_k \in N_t^s -} (1-\alpha^{k,s}) \right] \div \left[\sum_D + \sum_N + \sum_L \right] \quad (5)$$

For each cell $c_i \notin E_t^s$ (an Unobserved Cell), the conditional probability evolution is given below:

$$x_{t+1}^i = \left[x_t^i \times \prod_{c_j \in D_t^s} (\alpha^{j,s}) \times \prod_{c_k \in N_t^s} (1-\alpha^{k,s}) \right] \div \left[\sum_D + \sum_N + \sum_L \right] \quad (6)$$

with $\sum_D$, $\sum_N$, and $\sum_L$ defined as follows:

$$\sum_D = \sum_{c_i \in D_t^s} \left[x_t^i(1-\beta^{i,s}) \times \prod_{c_j \in D_t^s -} (\alpha^{j,s}) \times \prod_{c_k \in N_t^s} (1-\alpha^{k,s}) \right] \quad (7)$$

$$\sum_N = \sum_{c_i \in N_s^t} \left[x_t^i(\beta^{i,s}) \times \prod_{c_j \in D_t^s} (\alpha^{j,s}) \times \prod_{c_k \in N_t^s -} (1-\alpha^{k,s}) \right] \quad (8)$$

$$\sum_L = \sum_{c_i \notin E^s} \left[x_t^i \times \prod_{c_j \in D_t^s} (\alpha^{j,s}) \times \prod_{c_k \in N_t^s} (1-\alpha^{k,s}) \right] \quad (9)$$

Conditional Probability Evolution - Type II Problems. For Type II problems, measurements obtained from searching cell i will only affect the local conditional probability x_{t+1}^i for those cells that were observed, that is, where cell $i \in E_t^s$.

For each cell $c_i \in D_t^s$ (a Detect Cell), the conditional probability evolution is given below:

$$x_{t+1}^i = \frac{x_t^i(1-\beta^{i,s})}{x_t^i(1-\beta^{i,s}) + (1-x_t^i)(\alpha^{i,s})} \quad (10)$$

For each cell $c_i \in N_t^s$ (a Non-Detect Cell), the conditional probability evolution is given below:

$$x_{t+1}^i = \frac{x_t^i(\beta^{i,s})}{x_t^i(\beta^{i,s}) + (1-x_t^i)(1-\alpha^{i,s})} \quad (11)$$

For each cell $c_i \notin E_t^s$ (an Unobserved Cell), the conditional probability evolution is given below:

$$x_{t+1}^i = x_t^i \quad (12)$$

Cost Structure. In general DP formulations, the cost structure consists of two portions, transition costs and a terminal cost. The cost function is additive in that cost incurred at stage t accumulates over time. Since the primary DSN resource constraint is related to power, we established a cost structure for our DP formulation that is based on power consumption.

Transition Costs. For each stage t, we execute a test u_t and pay a cost $g_t(x_t, u_t, w_t) \geq 0$ for implementing the test. We identified two major power expenditures related to transition costs: sensing cost and transmission cost. The sensing cost is a function of the number of sensors that are in test u_t while the transmission cost is only incurred if a sensor that is in test u_t submits a sensor report to its assigned cluster head.

Terminal Cost. For our formulation of the problem, we will not impose terminal costs since the iterative decision process continues until we reach a termination state within an acceptable "Gamma Neighborhood." In addition, we have not imposed any costs as a function of time to completion (number of stages required).

Objective Function. Our objective is to determine the optimal sequence of sensor tests that will allow us to reach one of the termination states with minimum expected total cost. Since there is no clear upper bound on the number of stages required, we must plan over an infinite time horizon.

For an infinite horizon problem, the usual DP objective is defined as follows: given an initial state x_0, find a policy $\pi = \{\mu_0, \mu_1, \ldots\}$, where $\mu_t : X \mapsto U, \mu_t(x_t) \in U_t$, for all $x_t \in X, t = 0, 1, \ldots$, that minimizes the cost function

$$J_\pi(x_0) = \lim_{N\to\infty} E_{w_t}\left[\sum_{t=0}^{N-1} g_t(x_t, \mu_t(x_t), w_t)\right] \quad (13)$$

subject to the system equation constraint (3), repeated below for ease of reference.

$$x_{t+1} = f_t(x_t, u_t, w_t), \; t = 0, 1, \ldots.$$

Since termination appears to be inevitable for all reasonable heuristic policies π used in our ADP methodology, we make the assumption that the limit in the definition of $J_\pi(x_0)$ is finite. This assumption of inevitable termination allows us to consider the infinite horizon DSN sensor management problem as a finite (albeit random-length) horizon problem, where the length of the horizon is affected by the policy being implemented. The transformation from an infinite to finite horizon problem allows us to rephrase the cost function as follows:

$$J_\pi(x_0) = E_{w_t}\left[\sum_{t=0}^{T_\Omega} g_t(x_t, \mu_t(x_t), w_t)\right] \quad (14)$$

where we are totalling operating costs from stage $t = 0$ until stage T_Ω, the stage at which we first reach a termination state. The assumption of inevitable termination implies that T_Ω is finite with probability one and Equation 14 is well defined.

DP Algorithm. The DP algorithm for a finite horizon problem with fixed time horizon T reaching state x_T at stage T_Ω is defined via the following equations:

$$J_T(x_T) = g_T(x_T), \quad (15)$$

$$\begin{aligned} J_t(x_t) &= \min_{u_t \in U_t(x_t)} E_{w_t}\big[g_t(x_t, u_t, w_t) \\ &\quad + J_{t+1}(f_t(x_t, u_t, w_t))\big], \\ &\quad t = 0, 1, \ldots, T-1 \end{aligned} \quad (16)$$

Note that in (16), "min" denotes the greatest lower bound (or infimum) of the set of expectations over set $u_t \in U$. Even when the infimum is not attained, we will use the term "min" for convenience of notation.

On an infinite time horizon, using a total cost criterion, the optimal cost $J^*(x_0)$ for every state x_0 is equal to $J_0(x_0)$ given by the last step of the preceding DP algorithm, which proceeds backward in time from stage T_Ω to stage 0. The following limiting form of the DP algorithm should hold for all states x,

$$\begin{aligned} J^*(x) &= \min_{u \in U(x)} E_w\big[g(x, u, w) \\ &\quad + J^*(f(x, u, w))\big]. \end{aligned} \quad (17)$$

The expression above is a functional equation for the cost-to-go function J^*, and is called Bellman's equation (Bertsekas 2000). Bellman's equation is really a system of equations, one for each state, where the optimal expected cost-to-go from state x_t is coupled to the optimal expected cost-to-go from neighboring states x_{t+1}. In (17), the term $g(x, u, w)$ denotes transition costs incurred by transitioning from state x_t to state x_{t+1} and the term $J^*(f(x, u, w))$ denotes the optimal cost-to-go from state x_{t+1}.

Thus, to act optimally at state x, we need to choose action u that minimizes

$$E_w\big[g(x, u, w) + J^*(f(x, u, w))\big]. \quad (18)$$

In doing so, we choose actions that minimize the expected cost of a single transition plus the optimal expected long-term cost from the next state x_{t+1}. This will allow optimal actions to weigh both the short-term and long-term costs from state x_t. The optimization of this system of equations is nonlinear,

since we minimize with respect to u at each stage.

In very simple problems it is possible to obtain a closed-form solution to the DP algorithm presented above, but these cases tend to be in the minority. For most realistic problems, it is necessary to numerically solve the DP equations to obtain an optimal policy. For large-scale problems like the DSN sensor management problem, both the state and control spaces are very large. In these cases, the "curse of dimensionality" makes it difficult to compute J^* and hence infeasible to attempt a complete solution of the problem by DP. One approach to address this problem is to use various ADP techniques to approximate J^* and then use the approximation to construct a workable policy μ.

Solution Approximation. Policy iteration entails starting with an initial policy, evaluating the policy (policy evaluation step), and then deriving an improved policy (policy improvement step) (Bertsekas and Tsitsiklis 1996). We decided to use a single-stage lookahead rollout policy to execute the policy iteration.

Rollout Policy. For our policy evaluation step, we use Q-factors. These Q-factors consist of a state-control pair (x_t, u_t) and a stationary policy μ, defined as

$$Q_\mu(x_t, u_t) = \sum_{x_{t+1}} p_{x_t, x_{t+1}} \left[\underbrace{g_t(x_t, u_t, x_{t+1,i})}_{\text{transition cost}} + \underbrace{J_\mu(x_{t+1})}_{\text{cost-to-go}} \right] \quad (19)$$

The Q-factor denotes the expected cost corresponding to starting at state x_t, using control u_t at the first stage, and using the stationary policy μ at the second and subsequent stages.

In order to conduct the policy improvement step at each stage t, we select action $\mu(x_t)$, which represents the action corresponding to the minimum value Q-factor, using the equation:

$$\mu(x_t) = \min_{u \in U(x_t)} Q_\mu(x_t, u_t). \quad (20)$$

However, as was the case in trying to develop a closed-form, exact solution to the DP algorithm, we again run into problems trying to develop a closed-form, exact solution to equation (19). Since we have very large state and control spaces, we speed up the calculation of the rollout policy, but accept some potential performance degradation, by identifying a subset $\tilde{U}(x_t)$ of promising control actions to evaluate, rather than the full set of possible control actions.

Q-Factor Approximation. In our implementation of the single-stage lookahead rollout policy, we approximate the Q-factors by using simulation to approximate both the transition costs $g_t(x_t, u_t, w_t)$ and the cost-to-go functions $J^*(f_t(x_t, u_t, w_t))$ in (19).

For our stationary policy μ in (19) we use the Base Policy. That is, we execute the Base Policy μ_{BP} at the second and subsequent stages of our rollout policy. Even though the Base Policy is not optimal, it is useful since it is expected to always reach termination quicker than any other policy.

We then approximate the Q-factor for each candidate control action u_t by conducting K simulations of a transition from state x_t under test u_t to state $\hat{x}_{t+1}$ and then all subsequent transitions from state $\hat{x}_{t+1}$ under the

Base Policy μ_{BP} to termination.

We then use the vector of transition costs,

$$\left\{\hat{g}_t(x_t, u_t, \hat{x}_{t+1,1}), \hat{g}_t(x_t, u_t, \hat{x}_{t+1,2}), \ldots, \hat{g}_t(x_t, u_t, \hat{x}_{t+1,K})\right\} \quad (21)$$

and the vector of cost-to-go's,

$$\left\{\hat{J}_{\mu_{BP}}(\hat{x}_{t+1,1}), \hat{J}_{\mu_{BP}}(\hat{x}_{t+1,2}), \ldots, \hat{J}_{\mu_{BP}}(\hat{x}_{t+1,K})\right\} \quad (22)$$

to obtain an approximation of the Q-factor, as follows:

$$\hat{Q}_{\mu_{BP}}(x_t, u_t) = \frac{1}{K}\sum_{i=1}^{K}\left[\underbrace{\hat{g}_t(x_t, u_t, \hat{x}_{t+1,i})}_{\text{transition cost}} + \underbrace{\hat{J}_{\mu_{BP}}(\hat{x}_{t+1,i})}_{\text{cost-to-go}}\right] \quad (23)$$

The expected cost accumulated during each such trajectory is one estimate of the Q-factor. By simulating a large enough number of such trajectories, K, we can obtain an accurate approximation of the Q-factor, $\hat{Q}_{\mu_{BP}}(x_t, u_t)$, by averaging the results of all the trajectories.

Finally, comparing these Q-factor approximations, we implement the candidate control action $\overline{\mu}(x_t)$ corresponding to the minimum value Q-factor, or in other words,

$$\overline{\mu}(x_t) = \min_{u_t \in \hat{U}(x_t)} \hat{Q}_{\mu_{BP}}(x_t, u_t) \quad (24)$$

Results and Analysis

In this section we describe the Policy Comparison Analysis we conducted to evaluate various DSN sensor management policies and the Target Comparison Analysis we conducted to examine the impact of target location on system performance. During the Policy Comparison Analysis, we compared various dynamic control policies with the Base Policy and the SNOOPS Policy, while holding system parameters constant. During the Target Comparison Analysis, we compared various target configurations to determine how changes in target location affected the ability of the model to reach termination and the related operating costs.

During the simulation process, it was possible to collect a number of numerical measures of performance (MOPs) with which to evaluate the performance of the individual DSN sensor management policies. The comparison involved several key metrics, including: Total Cost, Number of Stages required to reach termination, Computation Time, and Success Rate.

Total Cost represents the cumulative DSN operating costs required to reach termination. The values presented represent the total units of cost to reach termination, both sensing costs and transmission costs. The Number of Stages required to reach termination is self-explanatory. Computation Time represents the amount of time required to complete the simulation run and should only be used to compare the magnitude of performance rather than specific performance

values. The values presented represent the number of seconds of computation time required to execute the simulation run. Success Rate represents the percentage of the replications that resulted in identifying the correct target location upon reaching termination.

Policy Comparison Analysis. We conducted a Policy Comparison Analysis to compare the performance of several heuristic dynamic control policies with the Base Policy and the SNOOPS Policy.

Design of Experiments. For this analysis, we used a repeated measures design. We conducted two iterations for the analysis, one for Type I problems and the other for Type II problems, so that we could compare model performance between the two processing techniques.

The subjects consisted of 500 random target locations for the Type I problem iteration and 1000 random target locations for the Type II problem iteration. These target locations were randomly selected based on the prior probability distribution.

The treatments consisted of seven different DSN sensor management policies, each conducted using the same set of sensor performance characteristics and model parameters (termination tolerance $\gamma = 0.1$ and false alarm rate (FAR) = 0.2) but each with a different random number seed. The seven treatments are described below:

- SNOOPS Policy: Use the simulation - based policy iteration ADP technique.
- Base Policy: Activate all sensors in the DSN.
- Action 1: Search the 10 cells with the highest x_t^i values, using the single best sensor for each cell.
- Action 2: Search the 20 cells with the highest x_t^i values, using the single best sensor for each cell.
- Action 3: Search the 30 cells with the highest x_t^i values, using the single best sensor for each cell.
- Action 4: Search the 20 cells with the lowest x_t^i values, using the single best sensor for each cell.
- Action 5: Search the 10 cells with the highest x_t^i values, using the two best sensors for each cell.

We expected that the Base Policy and SNOOPS Policy would provide an upper and lower bound on expected total operating costs. The Base Policy should provide an upper bound since it is expected to incur the greatest cost during each sensing iteration. The SNOOPS Policy should provide a lower bound since the simulation-based policy iteration technique is designed to select the control action from the set $\tilde{U}(x_t)$ that is "best" for each sensing iteration.

Type I Problem Results. The Total Cost results for the Type I problem simulation runs are depicted graphically in Figure 10. The figure represents the 95 percent confidence intervals for the mean total cost, calculated over 500 replications, each using a different random number seed.

Conducting ANOVA and using the Scheffé procedure at the 0.05 percent significance level, we determined that there was no significant difference between Action 5 and the SNOOPS Policy. However, these two policies were significantly better than the group

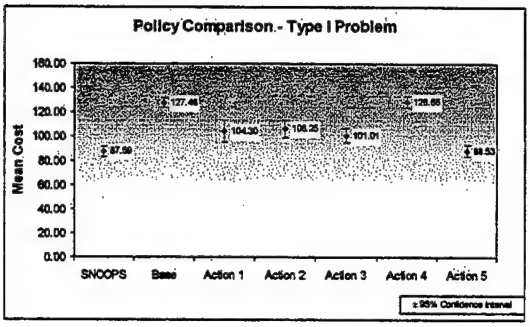

Figure 10: Type I Problem Results

consisting of Action 3, Action 1, and Action 2, which were not significantly different from each other. This second group was significantly better than the group consisting of the Base Policy and Action 4, which were not significantly different from each other.

Type II Problem Results. The Total Cost results for the Type II problem simulation runs are depicted graphically in Figure 11. The figure represents the 95 percent confidence intervals for the mean total cost, calculated over 1000 replications, each using a different random number seed.

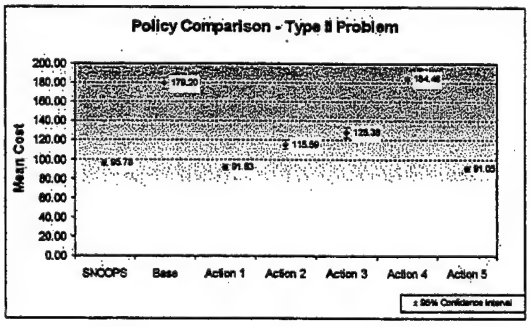

Figure 11: Type II Problem Results

Conducting ANOVA and using the Scheffé procedure at the 0.05 percent significance level, we determined that there was no significant difference between Action 5, Action 1, and the SNOOPS Policy. However, these three policies were significantly better than Action 2, which was significantly better than Action 3. This policy was in turn significantly better than the group consisting of the Base Policy and Action 4, which were not significantly different from each other.

Target Comparison Analysis. We conducted a Target Comparison Analysis to compare the performance of several heuristic dynamic control policies with the Base Policy and the SNOOPS Policy for specific target locations in order to determine the impact of target location on system performance.

Design of Experiments. For this analysis, we used a repeated measures design, just as for the previous analyses. We conducted two iterations for the analysis, one for Type I problems and the other for Type II problems, so that we could compare model performance between the two processing techniques.

The subjects consisted of 50 iterations for three target locations. To determine appropriate target locations, we reviewed the individual results from the Policy Comparison Analysis. We found that target locations fell into one of three categories: "easy", "average", and "difficult". For example, the four most difficult targets to detect and localize (i.e., resulting in the highest total costs) were Targets 6, 14, 35, and 47, as depicted in Figure 12.

We decided to choose one of these "difficult" targets, (Target 47), an "average" target (Target 1), and an "easy" target (Target 33) as the subjects for the Target Comparison Analysis.

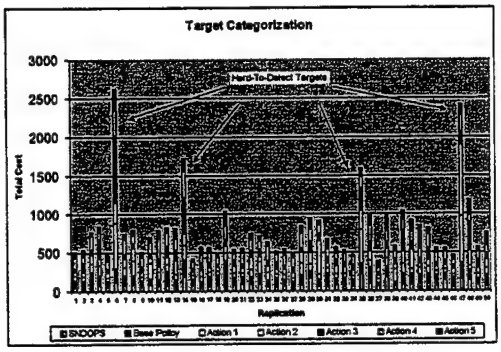

Figure 12: Target Location Difficulty

We used the same seven treatments as for the Policy Comparison Analysis, each conducted using the same set of sensor performance characteristics and model parameters ($\gamma = 0.1$, and FAR $= 0.2$) but each with a different random number seed.

Type I Problem Results. The Total Cost results for the Type I problem simulations are depicted graphically in Figures 13, 14, and 15. Each figure depicts the 95 percent confidence intervals for the mean total cost, calculated over 50 replications, each using a different random number seed.

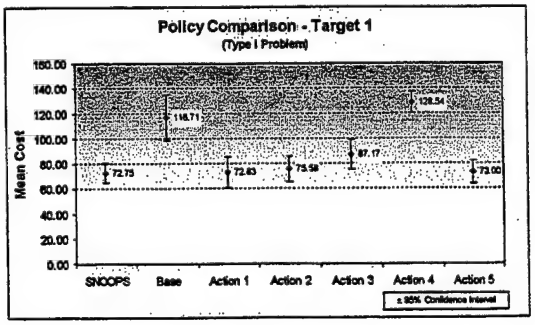

Figure 13: Type I Problem Results - Total Cost (Target 1)

Conducting ANOVA and using the Scheffé procedure at the 0.05 percent significance level, we determined that for Target 1 there was no significant difference between the SNOOPS Policy, Action 1, Action 5, Action 2, and Action 3. However, this group was significantly better than the group consisting of the Base Policy and Action 4, which were not significantly different from each other.

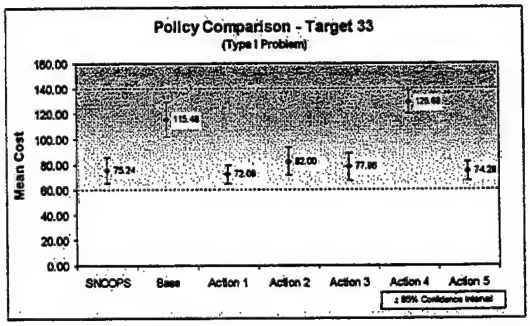

Figure 14: Type I Problem Results - Total Cost (Target 33)

Conducting the same analysis for Target 33, we found similar findings. There was no significant difference between the SNOOPS Policy, Action 1, Action 5, Action 2, and Action 3. However, this group was significantly better than the group consisting of the Base Policy and Action 4, which were not significantly different from each other.

Conducting the same analysis for Target 47, we found that there was no significant difference between the Base Policy, the SNOOPS Policy, Action 4, Action 5, Action 2, and Action 3. This group was significantly better than Action 1.

Type II Problem Results. The Total Cost results for the Type I problem simulations are depicted graphically in Figures 16, 17, and 18. Each figure depicts the 95 percent

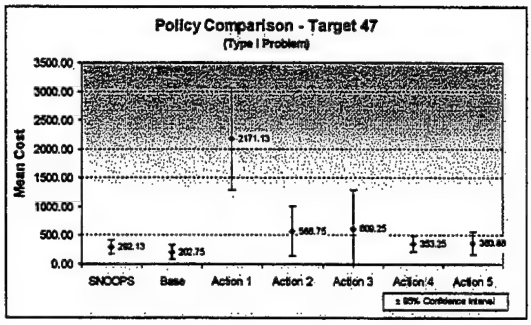

Figure 15: Type I Problem Results - Total Cost (Target 47)

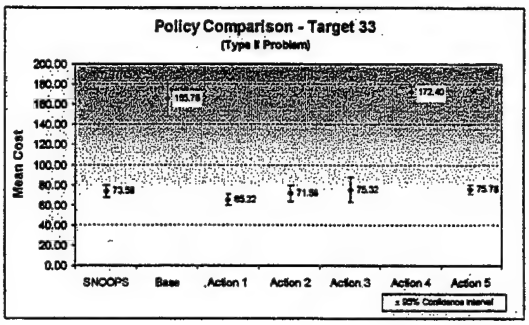

Figure 17: Type II Problem Results - Total Cost (Target 33)

confidence intervals for the mean total cost, each using a different random number seed.

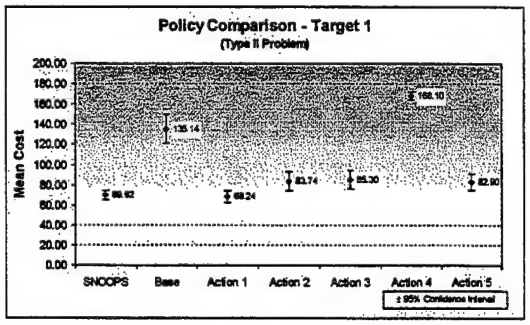

Figure 16: Type II Problem Results - Total Cost (Target 1)

Conducting ANOVA and using the Scheffé procedure at the 0.05 percent significance level, we determined that for Target 1, there was no significant difference between Action 1, the SNOOPS Policy, Action 5, Action 2, and Action 3. However, this group was significantly better than the Base Policy, which was significantly better than Action 4.

Conducting the same analysis for Target 33, we found that there was no significant difference between Action 1, Action 2, the SNOOPS Policy, Action 3, and Action 5. However, this group was significantly better than the group consisting of the Base Policy and Action 4, which were not significantly different from each other.

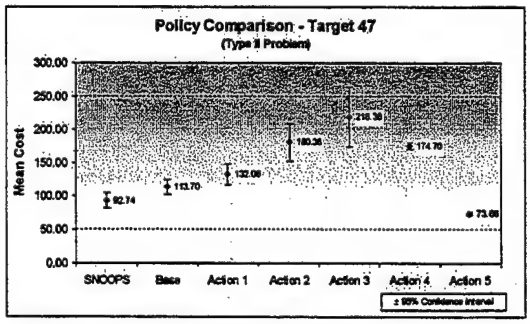

Figure 18: Type II Problem Results - Total Cost (Target 47)

Conducting the same analysis for Target 47, we found that Action 5 was significantly better than the group consisting of the SNOOPS Policy and the Base Policy, which were not significantly different from each other. This group was significantly better than Action 1, which was significantly better than the group consisting of Action

4, Action 2, and Action 3, which were not significantly different from each other.

Conclusions

In this paper, we have described our efforts to develop a model that can provide system-level management of a DSN. The goal of our research effort was to develop a model that could identify a sensor control strategy that could accomplish the sensing mission while reducing resource usage compared to the Base Policy of activating all sensors.

After reviewing the data, we are confident that our ADP approach is feasible for generating efficient DSN sensor management strategies for complex, large-scale DSNs. The sensor control strategy recommended by our model was more efficient than the Base Policy, requiring far less battery power to accomplish the same sensing mission. For Type I problems, the SNOOPS Policy used 31 percent less battery power than the Base Policy and for Type II problems, the SNOOPS Policy used 47 percent less battery power.

A comparison of the Base Policy with the SNOOPS Policy with all parameters held constant and the same target location is presented in Figure 19. While this comparison represents only a single instance, it is representative of the expected performance of both policies, based on the data.

In the specific instance represented by this figure, both the Base Policy and SNOOPS Policy were able to successfully locate the target. As expected, the Base Policy was quicker, taking only four stages versus six stages. However, again as expected, the Base Policy was more costly, consuming 175 units of power whereas the SNOOPS Policy only consumed 37 units. In this case, the

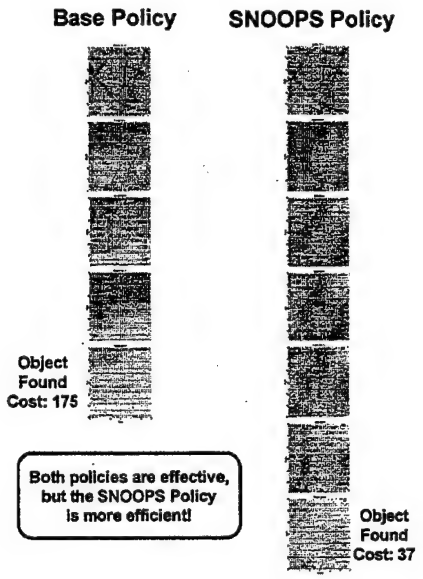

Figure 19: Base Policy vs. SNOOPS Policy

SNOOPS Policy was more efficient than the Base Policy.

Another important outcome of our research is the insight that simple dynamic control policies based on the underlying conditional probability distribution can be nearly as effective as the SNOOPS Policy while incurring a significantly lower computational burden.

Total Cost. These results indicate that the SNOOPS Policy indeed outperforms the Base Policy. As expected, the Base Policy generally provides an upper bound on expected total operating costs. For both Type I and Type II problems, the variability of the results make it difficult to identify any significant difference between the SNOOPS Policy and Actions 1, 2, 3, and 5. These results indicate that these four dynamic control policies perform comparably with the SNOOPS Policy in terms of total cost.

Number of Stages. These results were consistent for both Type I and Type II prob-

lems. In both cases, the Base Policy required the fewest number of stages to reach a termination state. For both Type I and Type II problems, the SNOOPS Policy and Actions 2, 3, and 5 were comparable in terms of how long it took to reach a termination state, generally two to three times longer than the Base Policy.

Computation Time. The general trend is that the SNOOPS Policy takes much longer than the Base Policy or any of the dynamic control policies. In fact, the computation time was on average about 80 percent more for the SNOOPS Policy than for the Base Policy. This is expected since there is a large amount of simulation required to execute the simulation-based policy iteration process that serves as the core of the SNOOPS Policy.

In comparing Type I and Type II problem computation times, we observed that the Type I problems took longer. This was a result of the assumption that there was a single target in the search region, requiring the update of the conditional probability for each cell in the search region with every execution of the Bayes update process. The Type II problems only updated observed cell conditional probability distributions.

Note that for the Type I problems the required computation time for the SNOOPS Policy averaged about 14 minutes, much too long to allow for real-time execution in a real world application. This was not a problem for the Type II problems since the average computation time was under a minute.

Success Rate. For Type I problems, each policy resulted in a 100 percent success rate in reaching the correct termination state, with the lone exception of Policy 1 for Target 47, where the success rate was 96 percent. For Type II problems, every policy exceeded an 82 percent success rate in reaching the correct termination state. The target that appeared to provide the most difficulty in terms of this measure was Target 1.

One source for the difference in results between the Type I and Type II problems could be the requirement to update every cell for the Type I problem. In this case, each observation provides more information since even the conditional probabilities for unobserved cells get updated. In the Type II problems it appears that the system is more likely to settle into an incorrect termination region, much akin to reaching a local optimum versus a global optimum.

SNOOPS Selection Rate. For Type I problems, Action 3 was selected the most often. Action 2 was selected the next most frequently, followed by Action 1, Action 5, and Action 4. The obvious outlier was Action 4, which was selected almost half as often as the other policies.

For Type II problems, Action 5 was selected the most often. Action 1 was selected the next most frequently, followed by Action 3, Action 2, and Action 4. The obvious outlier was Action 4, which again was selected almost half as often as the other policies.

Impact of Target Location. For both Type I and Type II problems, the DSN had difficulty in detecting Target 47 although the magnitude of difficulty was much higher for the Type I problem. It turns out that Target 47 is almost on top of a sensor, but still within range of several other sensors. Interestingly, this is also the case for the other identified "difficult" targets in Figure 12: Targets 6, 14, and 35. It appears that this combination of features prevents the DSN from quickly isolating the target location

and reaching a termination state.

Future Research

This work establishes a sound foundation for continued research into the DSN sensor management problem. However, there is a tremendous amount of work remaining in this area as we continue to improve the SNOOPS simulation model to make it more robust and versatile. In addition, there are possible extensions of this work to address other sensor issues as well as other research areas.

SNOOPS Improvements. While SNOOPS is already a fairly capable DSN simulation model, we have already identified a number of improvements that will extend its capabilities, to include:

Moving targets. Implement a "Target Movement Filter" or "Latency Filter" to extend results to moving targets.

Multiple targets. Conduct simulations with multiple targets to determine if the single target results continue to apply. The structure is already present in SNOOPS but has yet to be exploited.

Disparate sensors. Conduct simulations with disparate sensors to examine the benefit of sensor collaboration. The structure is already present in SNOOPS but has yet to be exploited.

ADP mechanism. Investigate alternative Candidate Actions for $\tilde{U}_t$. Try various improvements to the cost-to-go approximation process, to include increasing from a one-step lookahead to a multiple-step lookahead.

Cluster assignment. Examine a dynamic cluster assignment capability.

Cost structure. Examine the effect of introducing costs to account for the number of stages required to reach termination. Investigate the impact of a "Number of Stages" penalty cost to encourage quicker termination while still trying to minimize operating costs. This could be accomplished by implementing a weighting mechanism in the optimization process. Investigate the impact of a phased cost function, where the cost of using a sensor increases as it approaches the end of it's battery life. This approach could be expected to better preserve the average battery power within the network by avoiding the expenditure of all the use available from a single sensor.

Increased realism. Implement terrain, vegetation, and weather filters to represent the impact of these features on both observation and communication.

Extensions to Other Sensor Issues. The SNOOPS model provides a new capability to examine critical aspects of sensor fusion and DSN sensor management. Possible uses of the model include:

Sensor placement. Identify "optimal" sensor locations by implementing various sensor placement schemes as static, stationary sensor usage policies. Simulating each of these policies can help determine which policy (or sensor location scheme) provides the best results.

Sensor mix. Evaluate various sensor network compositions to gain insights into sensor mix issues.

Sensor fusion. Derive insights into the fusion of observations from different sensor types. Investigate the ability of non-imaging sensors to provide adequate situational awareness where "precision" emplace-

ment of more capable sensors is not possible.

DSN operational concepts. Develop operational concepts to better integrate DSN operations with user needs.

References

D.P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 1. Athena Scientific, Belmont, Massachusetts, 2nd edition, 2000.

D.P. Bertsekas and J.N. Tsitsiklis. *Neuro-Dynamic Programming.* Athena Scientific, Belmont, Massachusetts, 2nd edition, 1996.

D.A. Castañon. Optimal Search Strategies in Dynamic Hypothesis Testing. *IEEE Transactions on Systems, Man, and Cybernetics*, 25(7), July 1995.

D.A. Castañon. Approximate Dynamic Programming for Sensor Management. In *Proceedings of Conference on Decision and Control*, volume 3, pages 1202–1207, December 1997.

L.P. Clare, G.J. Pottie, and J.R. Agre. Self-Organizing Distributed Sensor Networks. In *Proceedings of SPIE 13th Annual International Symposium on Aerospace/Defense Sensing, Simulation, and Controls (AeroSense)*, Orlando, Florida, April 1999.

D. Cochran, D. Sinno, and A. Clausen. Source Detection and Localization Using a Multi-Mode Detector: A Bayesian Approach. In *Proceedings of the IEE International Conference on Acoustic, Speech, and Signal Processing*, Phoenix, Arizona, March 1999.

J.W. Hopkins, B.T. Mayes, D.B Hillis, H.Q Vu, and J.C. Brown. Warrior Extended Battlespace Sensors (WEBS). Technical report, U.S. Army Research Laboratory, Adelphi, Maryland, June 2000.

K. Kastella. Discrimination Gain to Optimize Detection and Classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 27, January 1997.

R. Krzysztofowicz and D. Long. Fusion of Detection Probabilities and Comparison of Multisensor Systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 20(3):665–677, May/June 1990.

R. Malhotra. Temporal Considerations in Sensor Management. In *Proceedings of the IEEE 1995 NAECON*, volume 1, pages 86–93, New York, May 1995.

G.A. McIntyre. An Information Theoretic Approach to Sensor Scheduling. In *Proceedings of the SPIE, Signal Processing, Sensor Fusion, and Target Recognition VI*, Orlando, Florida, April 1996.

G.A. McIntyre. *A Comprehensive Approach to Sensor Management and Scheduling.* PhD thesis, George Mason University, Fall 1998.

G.A. McIntyre and K.J. Hintz. Sensor Management Simulation and Comparative Study. In *Proceedings of the SPIE*, volume 3068, Orlando, Florida, April 1997.

R.L. Moses, R.M. Patterson, and W. Garber. Self Localization of Acoustic Sensor Networks. In *Proceedings of the 2002 MSS Specialty Group on Battlefield Acoustic and Seismic Sensing*, volume 1, Laurel, MD, 26-28 September 2002.

S.D. Patek. A Framework for Fault Isolation with Sensor Networks. 2001.

G.J. Pottie and W.J. Kaiser. Wireless Integrated Network Sensors. *Communciations of the ACM*, May 2000.

W. Schmaedeke. Information Based Sensor Management. In *SPIE Conference on Target Recognition and Data Fusion II*, Orlando, Florida, May 1993.

D. Sinno, D. Cochran, and D.R. Morrell. A Bayesian Risk Approach to Multi-Mode Detection. URL: citeseer.nj.nec.com/291447.html.

N. Srour. Unattended Ground Sensors - A Prospective for Operational Needs and Requirements. Technical report, U.S. Army Research Laboratory, October 1999.